Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK) DOI: 10.25126/jtiik.2022106409
Vol. 10, No. 2, April 2022, hlm. 291-300 p-ISSN: 2355-7699
Akreditasi KEMENRISTEKDIKTI, No. 36/E/KPT/2019 e-ISSN: 2528-6579

ANALISIS DAN PERANCANGAN APLIKASI CHATBOT MENGGUNAKAN FRAMEWORK RASA DAN SISTEM INFORMASI PEMELIHARAAN APLIKASI (STUDI KASUS: CHATBOT PENERIMAAN MAHASISWA BARU POLITEKNIK ASTRA)

Laksmi Anindyati[*1]

[1]Politeknik Astra, Jakarta Utara
Email: [1]laksmi.anindyati@gmail.com
*Penulis Korespondensi

(Naskah masuk: 01 Agustus 2022, diterima untuk diterbitkan: 10 April 2023)

Abstrak

Chatbot menjadi suatu kebutuhan bisnis yang membutuhkan pelayanan interaksi secara *real-time* dan 24 jam. Kebutuhan tersebut juga diperlukan saat penerimaan mahasiswa baru di Politeknik Astra. Chatbot dapat menjadi salah satu penyedia informasi yang interaktif untuk calon mahasiswa Politeknik Astra, ketika mencari informasi terkait proses pendaftaran mahasiswa baru maupun terkait Politeknik Astra secara umum. Proses analisis dan perancangan sistem dilakukan, dimulai dengan studi literatur. Hasil dari studi literatur dipilihlah Framework RASA yang akan digunakan dalam pengembangan chatbot. Framework RASA memiliki performa yang baik karena memiliki Rasa NLU dan Rasa CORE. Rasa NLU sebagai basis library yang membangunteraksi antara komputer dan manusia dengan menerapkan dua metode dan algoritma kecerdasan buatan yaitu pemrosesan bahasa alami dan mesin pembelajaran. Rasa NLU bertanggung jawab membuat interaksi lebih nyata, pengguna layanan akan merasakan interaksi langsung seperti dengan manusia bukan dengan komputer. Rasa CORE juga berperan dalam membuat interaksi terasa nyata, dengan mengatur interaksi dialog antara antara bot (komputer dibalik chatbot) dengan pengguna. Framework Rasa juga bersifat *open source* sehingga memiliki adaptabilitas yang tinggi ketika diperlukan modifikasi untuk menyesuaikan kebutuhan bisnis yang ada. Proses pengembangan sistem dilanjutkan dengan melakukan analisis dan perancangan sistem informasi penunjang. Sistem informasi penunjang chatbot ini dibangun untuk mengakomodir kebutuhan proses CRUD (*Create, Read, Update* dan *Delete*) dari pertanyaan–respon yang nantinya dipelajari oleh chatbot sebelum dapat berinteraksi seperti manusia. Sistem informasi penunjang ini membantu penyesuaian konfigurasi chatbot dalam merespon pertanyaan sehingga operasional kebutuhan tersebut dapat mudah dilakukan oleh admin tanpa latar belakang IT. Hasil dari penelitian ini adalah kebutuhan sistem yang direpresentasikan pada use case diagram dan flowchart lalu pemilihan pipeline NLU untuk chatbot, arsitektur sistem, perancangan database dalam bentuk physical data model, dan perancangan desain antarmuka (mockup) sistem penunjang chatbot framework RASA.

Kata kunci: *Open source Chatbot, Framework Rasa, Sistem Informasi, Analisis dan Perancangan Sistem Informasi, Kecerdasan Buatan*

ANALYSIS AND DESIGN CHATBOT APPLICATION USING RASA FRAMEWORK AND INFORMATION SYSTEM FOR MAINTENANCE CHATBOT APPLICATION

(CASE STUDY: CHATBOT ADMISSION OF NEW STUDENTS IN POLYTECHNIC ASTRA)

Abstract

Chatbots have become essential in business that requires interaction with customers in real-time and 24 hours. The requirements have become a necessity in Polytechnic Astra especially during the acceptance period of new students. Chatbot can be an interactive provider of information to prospective students of Polytechnic Astra who are looking information about the registration process or information related to Polytechnic Astra in general. The analysis and design process are conducted, starting with study literature. The results of the literature study, the RASA Framework were chosen as a tool to develop chatbots. RASA Framework performs well with the Rasa NLU and Rasa Core. Rasa NLU as a base library to build interactions between computers and humans using artificial intelligence. Rasa NLU is responsible for making interaction much real, like direct interaction with humans. Rasa core is a base library to regulate the interaction dialogue between chatbots and users. The Rasa Framework is

also open source, so it has high adaptability to be modified to suit existing business needs. This supporting information system help to adjust the configuration chatbot in responding to questions, so that the operational can be easily carried out by the admin without IT background. The results of this research are the system requirements represented in use case diagrams and flowcharts and the selection of NLU pipeline for chatbots, the system architecture, the database design in the form of physical data models, and the interface design (mockup) for the chatbot framework support system RASA.

Keywords: *Open source Chatbot, Framework Rasa, Information System, Analysis and Design Information System, Artificial Intelligence*

1. PENDAHULUAN

1.1. Latar Belakang

Sistem chatbot dengan basis AI saat ini banyak digunakan di berbagai jenis sector industry (seperti kesehatan, hukum, retail dan penididkan) untuk menunjang kebutuhan pelanggan, bahkan perkembangan baru-baru ini chatbot juga digunakan di tempat kerja untuk mendukung karywaan (Gkinko and Elbanna, 2022). Chatbot dengan basis AI dibangun untuk otomasi proses pelayanan (Janssen et al., 2022). Chatbot dianggap mampu mengurangi biaya *customer service* dan dapat lebih banyak melayani pelanggan secara *realtime* (Ranoliya, Raghuwanshi and Singh, 2017). Chatbot juga dapat meningkatkan kepuasan pelanggan (Mulyono, 2022). Pengembangan Chatbot Penerimaan Mahasiswa Baru (PMB) berangkat dari kebutuhan tersebut. Chatbot akan menjadi sebuah layanan yang menjadi gerbang utama untuk menjembatani interaksi dua arah antara calon peserta didik dan Politeknik Astra khususnya tim penerimaan mahasiswa baru.

1.2. Tujuan

Chatbot yang dikembangkan akan berperan sebagai *Virtual Assistan*t sederhana yang dapat menyediakan informasi spesifik terkait proses penerimaan mahasiswa baru dan terkait Politeknik Astra. Informasi yang akan disediakan oleh chatbot diantaranya adalah: syarat pendaftaran, biaya pendidikan, informasi beasiswa, program studi yang ditawarkan, pelayanan dormitory, dsb.

Dengan adanya chatbot maka memungkinkan adanya sebuah interaksi pelayanan kapanpun selama 24 jam dan dilakukan secara *realtime* tanpa perlu ada admin yang ikut mengontrol secara aktif (Ahmad, Hamid and Zainal, 2018; Mekni, 2021). Hal tersebut dapat meningkatkan *customer satisfaction* dan *customer experience*, dalam hal ini adalah calon peserta, saat membutuhkan sebuah informasi. Selain itu, secara umum chatbot juga akan mengurangi biaya operasional pelayanan dikarenakan tidak perlu adanya admin yang melayani interaksi antara calon peserta didik ketika mencari informasi.

1.3. Batasan Masalah

Batasan masalah dalam pengembangan sistem ini adalah:

1. Rasa framework digunakan sebagai sistem yang menunjang aplikasi chatbot PMB
2. Pertanyaan yang disiapkan adalah pertanyaan yang ada pada daftar *Frequently Question Answer* (FAQ)
3. Sistem penunjang aplikasi chatbot yang dirancang adalah sebuah sistem informasi yang dapat membantu memelihara data pertanyaan–jawaban. Seperti membuat data pertanyaan–jawaban baru, memperbarui atau menghapus data pertanyaan–jawaban yang sudah ada pada sistem.

2. METODE PENELITIAN

Metode penelitian yang dilakukan untuk pengembangan sistem informasi penunjang menyesuaikan dengan metode pengembangan perangkat lunak. Metode pengembangan perangkat lunak dikenal dengan istilah seperti *software process model* atau juga disebut sebagai *Software Development Life Cycle* yang selanjutnya akan disingkat menjadi SDLC. Metode SDLC yang dipilih untuk pengembangan sistem ini adalah adalah model waterfall.

Model waterfall adalah contoh penerapan dari prinsip *plan-driven*. Tahapan pada model waterfall menunjukan komponen fundamental aktivitas yang harus dilakukan pada masing–masing fase. Adapun tahapannya sebagai berikut(Sommerville, n.d.)

1. *Requirement analysis and definition*.

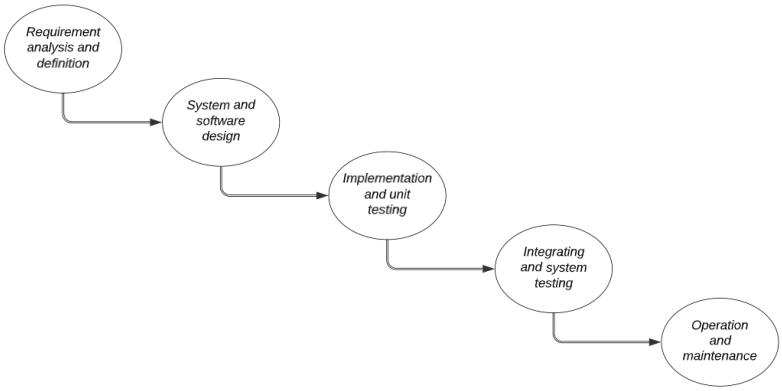
Mendefinisikan layanan dari sebuah sistem, batasan dan tujuan yang ditentukan dari kebutuhan pengguna sistem. Pada tahapan ini didefinisikan dengan detail spesifikasi dari sistem yang akan dikembangkan.

2. *System and software design*.

Pada *system and software design* dilakukan proses identifikasi dan penggambaran abstraksi fundamental dari sistem yang akan dibangun, beserta relasinya komponennya.

3. *Implementation and unit testing*.

Pada tahap ini desain dari *software* akan direalisasikan menjadi kumpulan program / unit program dan diujikan secara fungsi terkecil dari program.



Gambar 1. Diagram Model Waterfall

4. *Integrating and system testing*

Proses integrasi unit program dan pengujian sistem secara utuh dilakukan pada tahap ini. Tahap ini dilakukan untuk memastikan kebutuhan sistem yang didefinisikan sudah terpenuhi.

5. *Operation and maintenance*

Setelah sistem sudah terpasang dan sudah dioperasikan, berikutnya adalah tahapan *maintenance*. Tahapan *maintenance* adalah melakukan koreksi error yang pada tahapan sebelumnya tidak ditemukan. Perbaikan dari implementasi program / unit program dan perbaikan sistem juga dilakukan ketika kebutuhan baru dari sistem ditemukan.

3. KAJIAN LITERATUR

3.1. Chatbot

Chatbot adalah sebuah program komputer yang dapat menirukan percakapan manusia dengan menerapkan 2 metode dan algoritma *Artificial Intelligence*/kecerdasan buatan: Natural Language Processing dan Machine Learning (Adamopoulou and Moussiades, 2020; Caldarini, Jaf and McGarry, 2022). Chatbot merupakan salah satu bentuk dari *human–computer interaction* (HCI) yang menerapkan implementasi sistem kecerdasan buatan sehingga dapat meningkatkan pengalaman interaksi antara manusia dan komputer secara lebih baik. Istilah popular lainnya dari chatbot adalah *artificial conversation entity, interactive agents, smart bots, digital assistant, conversational system* (Bocklisch et al., 2017; Adamopoulou and Moussiades, 2020).

Terdapat dua pengelompokan chatbot berdasarkan kemampuan untuk merespon sebuah pertanyaan, yaitu: *open domain* dan *close domain*. *Open domain* chatbot dapat merespon pertanyaan umum tidak terbatas dengan topik tertentu, sedangkan *close domain* chatbot hanya dapat merespon pertanyaan yang sudah didefinisikan. Respon dari pertanyaan hanya berdasarkan pengetahuan tertentu yang topiknya sudah dibatasi (Nimavat and Tushar, 2017).

3.2. Framework Rasa

Framework Rasa pertama kali dikenalkan sebagai sebuah *dialogue management framework* level tinggi yang dapat menyederhanakan sebuah *conversational system* (Bocklisch et al., 2017). Framework Rasa dikembangkan berdasarkan pendekatan data / *data-based approach* (Harms et al., 2019). Framework Rasa dirasa mampu untuk menyamai performa dari *conversational system* komersial *data-based approach* lainnya seperti API.ai, IBM Watson dan LUIS. Dari penelitian komparasi yang pernah dilakukan antara API.ai, IBM, Watson dan LUIS ditemukan bahwa LUIS memiliki hasil output terbaik, meskipun begitu Rasa dapat mencapai nilai yang hampir sama dengan LUIS. Kesimpulan dari penelitian tersebut adalah jika Framework RASA dilakukan kostumisasi sangat memungkinkan mencapai hasil yang lebih baik dibandingkan framework sejenis (Braun and Langen, 2017).

Berbeda dengan *conversational system* komersial lainnya Framework Rasa bersifat *open source platform,* sehingga memiliki adaptabilitas yang tinggi dan kostumisasi dapat dilakukan. Tim pengembang dapat melakukan modifikasi sesuai dengan spesifikasi kebutuhan yang diinginkan atau disesuaikan dengan kebutuhan yang lebih spesifik. *Opens ource framework* terbukti lebih bisa dimodifikasi menyesuaikan kebutuhan pengembang dan dikendalikan secara lokal tanpa melibatkan pihak ketiga diluar organisasi (Malamas et al., 2021). Rasa juga sebagai merupakan framework yang cukup handal untuk kasus *low-resource language* sehingga sesuai jika diimplementasikan untuk Bahasa Indonesia(Windiatmoko, Hidayatullah and Rahmadi, 2009; Khan et al., 2021; Sholahuddin and Atqiya, 2021).

Cara Kerja Framework Rasa

Rasa dikembangkan menggunakan bahasa pemrograman Phyton, dan terdiri dari 2 library utama: Rasa Core dan Rasa NLU(Adamopoulou and Moussiades, 2020).

Rasa NLU adalah basis library yang akan memroses *natural language understanding* (NLU). Tujuan dari NLU adalah sebuah *library* yang digunakan untuk mengekstrak sebuah struktur dari sebuah input tidak terstruktur seperti sebuah kalimat chat, lalu memahami makna atau memahami secara semantik input tersebut (Nimavat and Tushar, 2017). NLU bisa disebut sebagai sebuah kumpulan kode untuk memahami *request* dalam bentuk kalimat chat yang masuk kedalam sistem chatbot untuk direspon lebih lanjut oleh bot. NLU ini berperan sebagai *interpreter* atau penerjemah sebuah chat. Dengan adanya NLU, chat dengan penggunaan kalimat berbeda dapat dipahami sebagai makna yang sama, berikut adalah contoh dari beberapa kalimat yang berbeda tetapi memiliki satu makna tujuan yang sama: "Tanya biaya kuliah di Politeknik Astra donk", "Biaya kuliah di Politeknik Astra berapa ya?", "Untuk kuliah di Politeknik Astra berapa biayanya?". Ketiga kalimat tersebut dapat dikenali dengan makna yang sama sebagai pertanyaan terkait biaya kuliah di Politeknik Astra.

Sebuah NLU dapat menerjemahkan maksud dari sebuah kalimat chat dengan melalui beberapa proses. Kalimat chat yang masuk akan diproses *tokenizing* lalu selanjutnya anotasi Parts of Speech (POS), pencarian representasi kata dilakukan menggunakan *word vector* dengan melihat keseluruhan kalimat, lalu melakukan klasifikasi menggunakan *supervised machine learning* untuk menemukan makna yang kemudian diterjemahkan menjadi sebuah *intent* tertentu (Adamopoulou and Moussiades, 2020). Proses ini, kedepannya disebut NLU Pipeline, dapat dikonfigurasi pada file config.xml.

Tuning NLU Pipeline dapat dilakukan untuk mendapatkan hasil yang paling baik. Secara umum gambaran proses yang diatur pada NLU Pipeline sebagai berikut:

1. *Tokenizing*: Proses memetakan sebuah kalimat dari sebuah karakter string menjadi sebuah kumpulan kata/*term* (Ridge, 1997). Secara sederhana proses *tokenizing* adalah memecah sebuah kalimat menjadi potongan text atau kedepannya disebut token.
2. *Featurizers*: Mengubah token menjadi sebuah fitur yang dikenali dan dapat diproses oleh *machine learning*
3. *Intent Classifier* dan *Entity Extraction*: Memroses fitur input untuk diklasifikasikan menjadi *intent* tertentu. Outputnya berupa rekomendasi *intent* dengan penentu probabilitas, atau komunitas pengembang Framework Rasa menyebutnya sebagai nilai *confidence*.

Rasa CORE adalah bagian library yang akan memproses manajemen dialog. Rasa CORE melakukan prediksi respon apa yang akan dikeluarkan oleh bot, dengan mempertimbangkan informasi dari histori percakapan yang sudah dilakukan sebelumnya. Fitur standar yang dilakukan oleh Rasa CORE ini adalah mempertimbangan percakapan yang dilakukan sebelumnya, melihat respon yang cocok, belajar dari percakapan dengan pengguna lain sebelumnya, dan melihat respon apa yang saat ini didefinisikan (Adamopoulou and Moussiades, 2020). Rasa CORE memastikan dialog interaktif dan dapat disesuaikan dengan kebutuhan. Sebagai contoh ketika ada *request* chat untuk melakukan pendaftaran maka urutan aturan respon yang perlu dilakukan oleh bot adalah meminta pengguna mengisi form pendaftaran, meminta pengguna untuk mentransfer uang pendaftaran, meminta pengguna untuk *upload* bukti pembayaran pendaftaran lalu respon terakhir adalah memberikan pernyataan atau bukti bahwa pendaftaran berhasil dilakukan. Urutan dialog tersebut diatur pada Rasa Core.

Semua data latih yang digunakan oleh Rasa NLU dan Rasa CORE disimpan dalam bentuk file YAML sehingga bisa dikelola oleh pengguna. YAML merupakan sebuah file bertipe .yml yang mudah dibaca oleh manusia sebagai pengguna dan mudah di-*parsing* sehingga cepat dipahami secara *syntax* oleh komputer. Ada 3 buah file YAML utama yang perlu disesuaikan dengan kebutuhan sistem yang akan dikembangkan yaitu domain.yml, stories.yml dan nlu.yml. Pada ketiga file tersebut ada beberapa *key* yang akan menjadi fokus untuk dikembangkan. <u>*Key*</u> tersebut adalah:

• story

story adalah sebuah *key* yang mendefinisikan urutan respon yang harus dilakukan sebuah chatbot. Contoh jika sebuah percakapan dimulai dari menanyakan tanggal lahir, berikutnya adalah bulan lahir dan tahun lahir maka urutan *respon* chatbot tersebut didefinisikan dalam satu buah story dengan urutan langkah tersebut. Secara sederhana story memetakan intent dengan action. Pendefinisian story disimpan pada stories.yml.

• action

action berisi sebuah respon yang berelasi dengan text tertentu. Text ini adalah pesan yang akan dikeluarkan oleh chatbot. Kumpulan action dan text disebut responses dan disimpan pada domain.yml

• intent

intent ini adalah sebuah *key* untuk mengklasifikasikan maksud dari kalimat yang dikirim user kepada chatbot atau *request chat*. Sebuah intent dipetakan dengan examples. Examples adalah beberapa contoh input chat *request* yang secara kalimat berbeda tetapi bermakna sama. Data intent dan examples ini ini disimpan pada nlu.yml

Konfigurasi data latih di ketiga file utama tersebut dapat dijadikan dasar untuk pembuatan sistem informasi yang mengintegrasikan rasa dengan sebuah aplikasi *web-based* sebagai penunjang.

4. HASIL DAN PEMBAHASAN

4.1 Analisis Sistem Penunjang Chatbot

Sistem penunjang chatbot merupakan sebuah sistem informasi yang dirancang dengan tujuan mempermudah pemeliharaan chatbot. Sistem penunjang chatbot ini merupakan sebuah sistem yang dapat melakukan konfigurasi framework Rasa menyesuaikan kebutuhan atau perubahan bisnis. Sebagai contoh, jika terjadi perubahan informasi seperti: biaya pendaftaran, syarat pendaftaran, penambahan prodi hal tersebut dapat dilakukan melalui sistem ini.

Saat proses analisis, ditemukan kebutuhan sebuah tempat penyimpanan / storage yang bersifat dinamis dan dapat diakses dengan mudah oleh pengguna. Tempat penyimpanan/ storage tersebut adalah sebuah database yang dapat diakses melalui sebuah aplikasi berbasis web dengan tampilan *user-friendly.*

Hasil analisis framework rasa, terdapat 3 buah file bertipe .yml yang berfungsi untuk mengatur respon dari chatbot, yaitu: NLU.yml, Stories.yml dan Domain.yml. Aplikasi penunjang menghubungkan konfigurasi tersebut dengan sebuah database lalu akan secara otomatis menghasilkan ketiga file tersebut dengan *trigger* pengaturan waktu.

Kebutuhan tersebut direpresentasikan menjadi sebuah use case diagram dengan dua buah aktor: Admin dan System Timer. Gambar 1 merupakan use case diagram dari sistem penunjang chatbot.

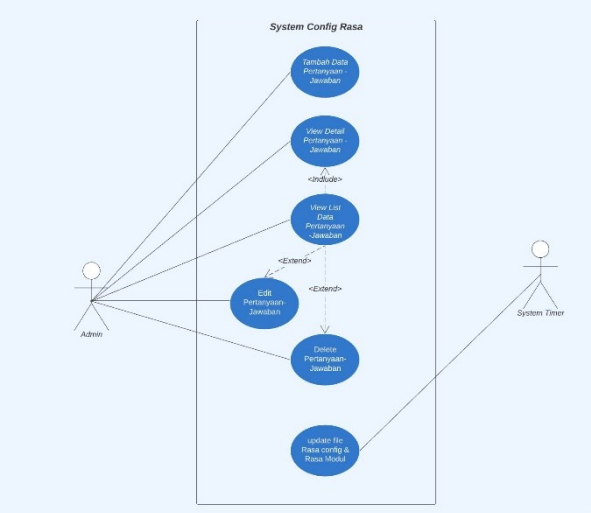
Jika dilihat dari pengguna sistem penunjang aplikasi chatbot maka terdapat 2 buah sistem utama yang tidak saling berkaitan secara langsung yaitu:

1. System Config Rasa–CRUD Data

System Config Rasa–CRUD Data berhubungan dengan aktor/user Admin. Sebelum masuk ke dalam sistem diperlukan proses login. Sistem ini memastikan respon chabot dapat dipelihara dengan menyediakan CRUD–Create, Read, Update dan Delete data pertanyaan–jawaban pada database, dan kemudian akan mempengaruhi cara chatbot memberikan respon sebuah pertanyaan.

Terdapat 5 fitur untuk memenuhi kebutuhan tersebut:

a. Tambah Data Pertanyaan–Jawaban
   Fitur ini berangkat dari kebutuhan untuk *create data* atau menambah data pertanyaan–jawaban.
b. View List Data Pertanyaan–Jawaban
   Kebutuhan untuk melihat keseluruhan data pertanyaan–jawaban dapat diakomodasi dengan fitur ini.
c. View Detail Data Pertanyaan–Jawaban
   Sistem tidak hanya menampilkan daftar pertanyaan dan jawaban, tetapi user juga dapat melihat detail dari pertanyaan–jawaban yang sudah tersimpan di dalam database melalui fitur *view detail* data pertanyaan - jawaban.



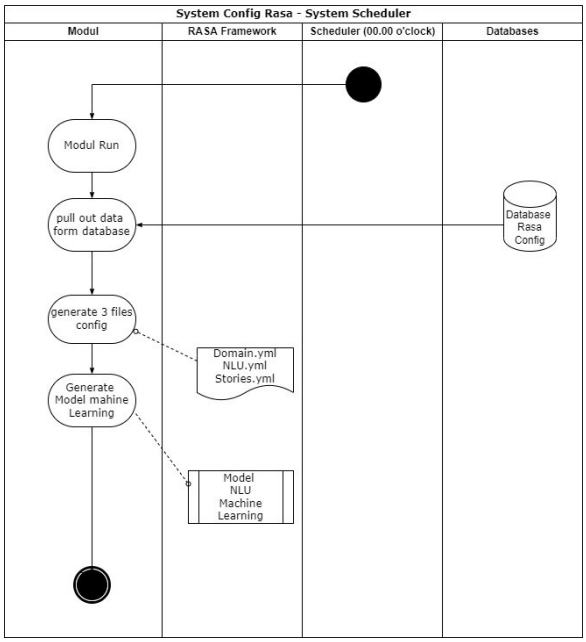
Gambar 2. Use case Diagram System Config Rasa

d. Edit Data Pertanyaan–Jawaban
   Fitur edit data pertanyaan-jawaban digunakan untuk kebutuhan modifikasi data pertanyaan–jawaban yang sudah ada pada database.
e. Delete Data Pertanyaan–Jawaban
   Fitur delete data pertanyaan–jawaban digunakan untuk kebutuhan menghapus data pertanyaan–jawaban yang tidak diperlukan lagi untuk chabot.

2. System Config RASA - System Scheduler

Sistem ini berangkat dari kebutuhan adanya sebuah pemicu/ *trigger* yang memungkinkan model dari framework dapat diperbaharui. Sistem ini merupakan aplikasi *standalone* yang ditanam pada server. Fitur utama dari sistem ini adalah *scheduler/*pengaturan waktu, *generate* 3 file utama yang mempengaruhi respon dari chatbot dan melakukan pelatihan *machine learning* untuk membuat model baru untuk digunakan framework Rasa.

Sistem akan mengeksekusi sebuah modul berdasarkan waktu tertentu, waktu akan diset pada pukul 00.00 setiap hari. Waktu tersebut dipilih dengan mencari waktu yang umumnya tidak banyak pengguna mengakses chatbot. Ketika modul dieksekusi, maka sistem akan menarik data dari database, lalu meng-generate 3 file konfigurasi utama: NLU.yml, Stories.yml dan Domain.yml. Setelah 3 file tersebut dibuat, modul akan melakukan *training* ulang untuk menciptakan sebuah model *machine learning* yang digunakan chatbot untuk mengklasifikasikan intent. Secara umum bisnis proses dari *System Scheduler* ini dapat dilihat pada flowchart gambar 2



Gambar 3. Flowchart System Scheduler

4.2. Perancangan

4.2.1 Perancangan Pipeline NLU

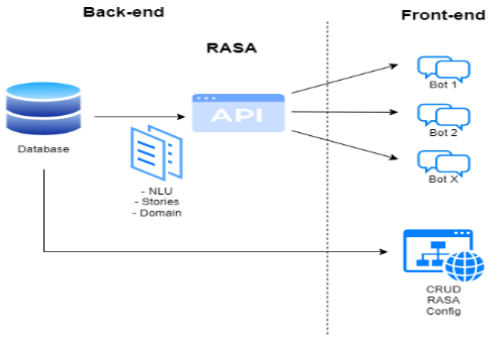
Seperti yang sudah dijelaskan sebelumnya pada landasan teori, Framework RASA dapat mengungguli beberapa framework komersial dengan dilakukan kustomisasi. Salah satu kostumisasi yang akan dilakukan adalah konfigurasi Pipeline NLU. Penelitian sebelumnya melakukan komparasi untuk menemukan *tuning* pipeline NLU yang paling baik (Khan et al., 2021). Adapun kombinasi pipeline NLU sebagai berikut:

"LanguageModelFeaturizer (BERT) + RegexFeaturizer + LexicalSyntacticFeaturizer + CountVectorFeaturizer + DIETClassifier + EntitySynonymMapper + FallbackClassifies"

Konfigurasi pipeline tersebut dianggap paling baik dikarenakan menggunakan BERT. BERT dinyatakan sebagai model pre-trained yang menjadi *state-of-the-art* untuk *text retrieval* karena secara performa dapat mengungguli model lainnya. BERT akan memecah token menjadi sebuah representasi fitur tertentu sebelum dilanjutkan ke proses pipeline berikutnya. Kombinasi konfigurasi pipeline tersebut akan dijadikan dasar utama untuk design pipeline NLU pada sistem chatbot yang akan dikembangkan.

4.2.2 Arsitektur Sistem

Arsitektur yang dibangun mempertimbangan konsep client server yang dibagi menjadi dua perspektif, yaitu: aplikasi backend dan aplikasi frontend. Gambar 3 adalah perancangan arsitektur dari sistem penunjang chatbot dan interaksinya dengan rasa framework. Aplikasi penunjang pada front-end memberikan tampilan pengguna yang user-friendly untuk system config rasa–CRUD dan juga tampilan agen chatbot yang berinteraksi dengan pengguna. Sedangkan aplikasi penunjang pada back-end berkaitan dengan database yang mempengaruhi konfigurasi dari framework RASA dan komunikasi request–response agen bot melalui API framework RASA.



Gambar 4. Arsitektur Sistem

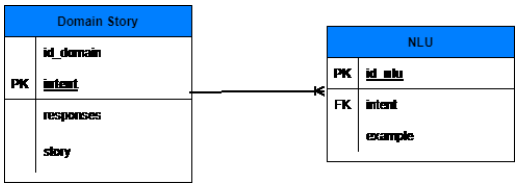
4.2.3 Perancangan Database: PDM

Perancangan database sebagai penyimpanan data dalam sistem penunjang chatbot digambarkan dalam bentuk Physical Data Model pada gambar 4. Kebutuhan 3 file konfigurasi Rasa dipetakan ke dalam bentuk PDM, dimana 1 field / data pada tabel merepresentasikan 1 kebutuhan key / value pada file yml. Detail pemetaan data file yml (key: value) sebagai berikut:

a. stories.yml disediakan pada tabel domain_story.
   - "story": gabungan string "pertanyaan" + spasi + <atribut intent>
   - "intent": <atribut intent>
   - "action": gabungan string "utter_" + <atribut intent>

b. domain.yml disediakan pada tabel domain_story
   - gabungan string "utter_" + <atribut intent>: <atribut responses>

c. nlu.yml disediakan pada table domain_story dan table NLU
   - "intent": <atribut intent> pada domain story
   - examples": daftar atribut example hasil select dari table NLU berdasarkan intent tertentu.

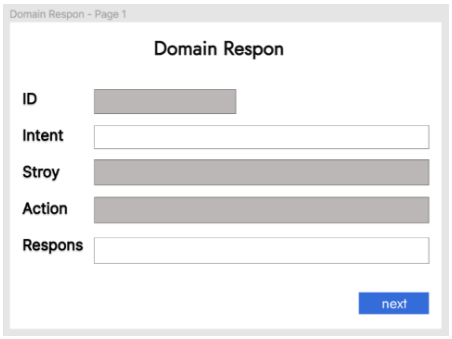


Gambar 5. Physical Data Model Database

4.2.4 Perancangan User Interface Sistem Penunjang

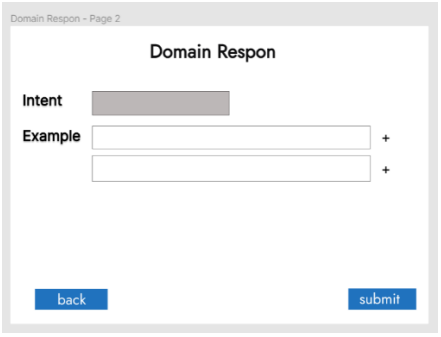
Perancangan User Interface berangkat dari kebutuhan fitur pada System Config Rasa–CRUD Data.

a. Menu Tambah Data Pertanyaan–Jawaban



Gambar 6. User Interface Add Domain Respon

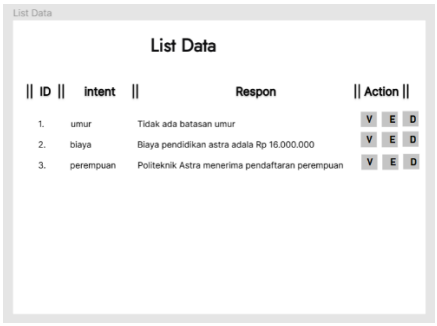
- ID: diisi otomatis berdasarkan data terakhir pada database
- Intent: text max 20, tidak boleh ada spasi
- Story: diisi otomatis "pertanyaan" + spasi + intent
- Action: diisi otomatis "utter_" + intent
- Respon: text max 500. Berisi respon dari intent
- Next: Tombol untuk ke page 2



Gambar 7. User Interface Add Domain Respon Page 2

- Intent: diisi otomatis berdasarkan data page1
- Example: text max 200, berisi data latih dari pertanyaan terkait intent
- +: tombol untuk menambah data example
- Back: tombol untuk kembali ke page 1
- Submit: tombol untuk menyimpan data ke dalam database

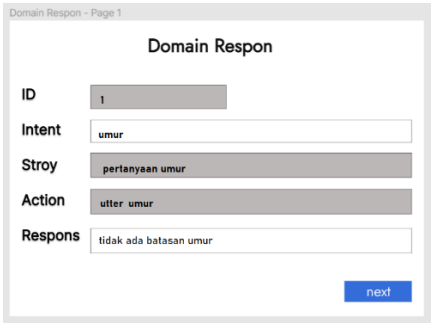
b. Menu View List Pertanyaan–Jawaban



Gambar 8. User Interface List Data Pertanyaan - Jawaban

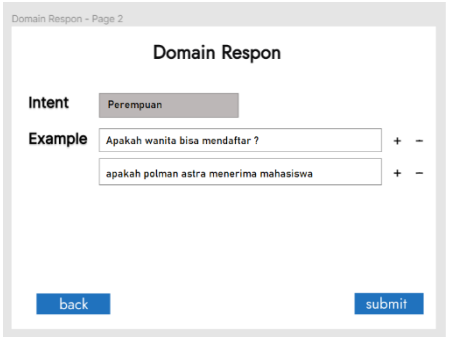
- V: tombol untuk masuk ke dalam fitur View Detail Data Pertanyaan–Jawaban
- E: tombol untuk masuk ke dalam fitur Edit Data Pertanyaan–Jawaban
- D: tombol untuk menghapus keseluruhan data yang berelasi dengan intent tersebut

c. Menu View Detail/Edit Data Pertanyaan–Jawaban



Gambar 9. User Interface View/Edit Domain Respon Page1

- ID: diisi otomatis berdasarkan data terakhir pada database
- Intent: view atau edit. Untuk edit text max 20, tidak boleh ada spasi
- Story: diisi otomatis: "pertanyaan" + spasi + intent
- Action: diisi otomatis: "utter_" + intent
- Respon: view atau edit. Untuk edit text max 500. Bersisi respon dari intent



Gambar 10. User Interface View/Edit Domain Respon Page2

- Intent: diisi otomatis berdasarkan data page1
- Example: View atau Edit. Untuk edit text max 200, berisi data latih dari pertanyaan terkait intent
- +: tombol untuk menambah data example (hanya ada pada menu edit)
- -: tombol untuk menghapus data example (hanya ada pada menu edit)
- Back: tombol untuk kembali ke page 1
- Submit: tombol untuk menyimpan data ke dalam database (hanya ada pada menu edit)

5. KESIMPULAN

Proses Analisis dan Perancangan Aplikasi Chatbot merupakan dua tahapan awal dalam proses model waterfall pada *Software Development Life Cycle* (SDLC). Kedua tahapan proses tersebut dilakukan untuk menemukan kebutuhan sistem dan perancangan untuk aplikasi chatbot dan sistem penunjang chatbot penerimaan mahasiswa baru Politeknik Astra.

Pada proses analisis kebutuhan sistem, dilakukan studi literatur untuk menemukan teknologi yang tepat. Hasil studi literatur menyimpulkan bahwa teknologi yang akan diterapkan adalah Framework RASA. Framework RASA dianggap sebagai sebuah framework *dialog management* terbaik dibandingkan dengan teknologi komersial lainnya dengan mempertimbangan desain dan fitur yang dimiliki. Framework Rasa dapat menghasilkan percakapan yang cukup baik karena memiliki 2 library utama yang sudah dapat menunjang natural language understanding (NLU RASA) dan pengaturan manajemen dialog (CORE RASA) dengan menerapkan *machine learning*. Rasa Framework juga bersifat *open source* sehingga kostumisasi menyesuaikan kebutuhan dapat dilakukan.

Tahapan analisis dilanjutkan untuk menemukan kebutuhan pengaturan pertanyaan - jawaban yang perlu dipelajari oleh chatbot. Atas dasar itu pula dilakukan analisis lebih lanjut untuk proses penunjang sistem chatbot yang dapat mengakomodasi proses CRUD (Create, Read, Update dan Delete) pertanyaan - jawaban chatbot. Proses penunjang sistem chatbot akan memelihara 3 file data utama yang berkaitan dengan data pertanyaan - jawaban yaitu: nlu.yml, stories.yml dan domain.yml. Hasil dari tahapan analisis adalah kebutuhan sistem yang digambarkan menggunakan usecase diagram dan juga flowchart. Tahapan Perancangan sistem dilakukan dengan merancang pipeline NLU, arsitektur sistem, perancangan database dalam bentuk physical data model, dan perancangan desain antarmuka (*mockup*) sistem.

Penelitian berikutnya dilakukan untuk melanjutkan implementasi sistem dan pengujian, selain itu saat dapat didalami kembali pipeline NLU untuk chatbot menggunakan Bahasa Indonesia dengan melakukan komperasi beberapa rancangan pipeline NLU.

DAFTAR PUSTAKA

ADAMOPOULOU, E. & MOUSSIADES, L., 2020. Machine Learning with Applications Chatbots : History , technology , and applications. *Machine Learning with Applications*, [online] 2(July), p.100006. https://doi.org/10.1016/j.mlwa.2020.100006.

AHMAD, N.A., HAMID, M.H.C. & ZAINAL, A., 2018. Review of Chatbots Design Techniques. *International Journal of Computer Applications*, 181(August).

BOCKLISCH, T., FAULKNER, J., PAWLOWSKI, N. & NICHOL, A., 2017. Rasa : Open Source Language Understanding and Dialogue Management. pp.1–9.

BRAUN, D. & LANGEN, M., 2017. Evaluating Natural Language Understanding Services for Conversational Question Answering Systems. (August), pp.174–185.

CALDARINI, G., JAF, S. & MCGARRY, K., 2022. A Literature Survey of Recent Advances in Chatbots. *Information*, 1, pp.1–24.

GKINKO, L. & ELBANNA, A., 2022. The appropriation of conversational AI in the workplace : A taxonomy of AI. *International Journal of Information Management*. [online] https://doi.org/10.1016/j.ijinfomgt.2022.102568.

HARMS, J., KUCHERBAEV, P., BOZZON, A. & HOUBEN, G., 2019. Approaches for Dialog Management in Conversational Agents. *IEEE Internet Computing*, 23, pp.13–22. https://doi.org/10.1109/MIC.2018.2881519.

JANSSEN, A., RODRÍGUEZ, D., PASSLICK, J. & BREITNER, M.H., 2022. How to Make chatbots productive–A user-oriented implementation framework. *International Journal of Human - Computer Studies*, [online] 168(September), p.102921. https://doi.org/10.1016/j.ijhcs.2022.102921.

KHAN, F.S., MUSHABBIR, M. AL, IRBAZ, M.S. & NASIM, A. AL, 2021. End-to-End Natural Language Understanding Pipeline for Bangla Conversational Agents. *International Conference on Machine Learning and Applications*. https://doi.org/https://doi.org/10.48550/arXiv.2107.05541.

MALAMAS, N., SYMEONIDIS, A., MALAMAS, N. AND SYMEONIDIS, A., 2021. ScienceDirect Embedding Embedding Rasa Rasa in in edge edge Devices : Devices : Capabilities Capabilities and and Limitations Limitations. *Procedia Computer Science*, [online] 192(2019), pp.109–118. https://doi.org/10.1016/j.procs.2021.08.012.

MEKNI, M., 2021. An Artificial Intelligence Based

Virtual Assistant Using Conversational Agents. *Journal of Software Engineering and Applications*, 14, pp.455–473. https://doi.org/10.4236/jsea.2021.149027.

MULYONO, J.A., 2022. Evaluation of Customer Satisfaction on Indonesian Banking Chatbot Services During the COVID-19 Pandemic. *CommIT Journal 16(1),* 16(1), pp.69–85.

NIMAVAT, K. AND TUSHAR, P., 2017. Chatbots : An Overview Types , Architecture , Tools and Future Possibilities. 5(07), pp.1019–1024.

RANOLIYA, B.R., RAGHUWANSHI, N. AND SINGH, S., 2017. Chatbot for university related FAQs. In: *2017 International Conference on Advances in Computing,* Communications *and Informatics (ICACCI)*. Udupi.pp.1525–1530. https://doi.org/10.1109/ICACCI.2017.8126057.

RIDGE, K., 1997. Critical Tokenization and its Properties. *Association For Computational Linguistics*.

SHOLAHUDDIN, M.R. AND ATQIYA, F., 2021. Sistem Tanya Jawab Konsultasi Shalat Berbasis RASA Natural Language Understanding (NLU). *Jurnal* Pendidikan *Multimedia*, 3(2), pp.93–102. https://doi.org/10.17509/edsence.v3i2.38732.

SOMMERVILLE, I.,2017 *Software Engineering*. 9th ed. Pearson.

WINDIATMOKO, Y., HIDAYATULLAH, A.F. AND RAHMADI, R., 2009. Developing FB Chatbot Based on Deep Learning Using RASA Framework for University Enquiries. nternational *Conference on Information Technology and Digital Applications*. https://doi.org/https://doi.org/10.1088/1757-899X/1077/1/012060.

Halaman ini sengaja dikosongkan